CHALMERS
UNIVERSITY OF TECHNOLOGY

Multi-Objective Optimization of a Counter Rotating Open Rotor using Evolutionary Algorithms

Multi-Objective Optimization of an Counter Rotating Open Rotor using Evolutionary Algorithms

Gonzalo Montero Villar*, Daniel Lindblad† and Niklas Andersson‡
Chalmers University of Technology, Gothenburg, SE-412 96

In the present work, a recently developed platform for turbomachinery design and optimization based on Evolutionary Algorithms (EAs) is presented. Two types of evolutionary algorithms are implemented. The first one based on Genetic Algorithms (GAs) and the second one on Differential Evolution (DE), both able to handle single- and multi-objective as well as constrained and unconstrained optimization problems. Meta-modeling based on Radial Basis Functions (RBFs) is used in order to help accelerate the optimization when the objective function is too expensive to be evaluated inside the EA. Details on the implementation as well as validation results for a set of well-known benchmark cases are presented. The platform is also combined with 3D CFD simulations to optimize the aerodynamic performance of a Counter Rotating Open Rotor (CROR). Results from the CROR optimization are discussed and analyzed.

I. Nomenclature

EA	evolutionary algorithm
GA	genetic algorithm
DE	differential evolution
RBF	radial basis function
LHS	latin hypercube sampling
$CROR$	counter rotating open rotor
$NSGA-II$	non-dominated sorting genetic algorithm II
cd	crowded distance
T	total thrust, N
P	total power, W
η	propeller efficiency
C_T	thrust coefficient
n	blade rotational speed, rev/s

II. Introduction

Evolutionary algorithms (EAs) are a class of optimization methods inspired by biological evolution that try to mimic the natural process to improve a population. These algorithms fall inside the category of stochastic optimization methods, which as opposed to conventional optimization techniques, such as gradient descent or quasi-Newton methods, do not dependent on the ability to compute derivatives of the function to be optimized. Since the introduction of Genetic Algorithms [1] (GAs) and Differential Evolution [2] (DE), both of which belong to the category of evolutionary algorithms, their popularity has increased due to their non-deterministic nature and capability of dealing with noisy discontinuous objective functions. Lots of progress has been done for the past decades working towards more efficient and versatile algorithms for both single- and multi-objective optimization as well as constrained and unconstrained problems [3, 4].

*Ph.D. Student, Division of Fluid Dynamics, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden

†Ph.D. Student, Division of Fluid Dynamics, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden

‡Assoc. Professor, Division of Fluid Dynamics, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden

With the increase in computational power, design and optimization methods based on EAs coupled with meta-models where several hundreds or thousands of design evaluations are performed are now feasible. Extensive work has been done in optimization of different turbomachinery components using evolutionary algorithms and meta-models. Ellbrant et al. [5] optimized a transonic compressor for maximum static pressure recovery and minimum pressure loss using multi-objective GA coupled with meta-models. They compared several meta-models, namely; 2nd order polynomial, Kriging-based, Radial Basis Functions (RBFs) and feedforward neural networks. They concluded that RBF was the best choice for their case. Marinus et al. [6] performed a multidisciplinary aerodynamic, aeroacoustic and aeroelastic optimization of transonic propeller blades with multi-objective DE and a Kriging-based meta-model. Schnell et al. [7] used multi-objective DE to optimize a counter rotating open rotor (CROR) configuration for low noise and high efficiency.

In this work an automated platform for turbomachinery design and optimization, called HAMON, is presented. It uses either GA or DE as the optimization method and has the capability of handling single- and multi-objective as well as constrained and unconstrained optimization problems. RBFs are also available to help accelerate the optimization process in case the evaluation of the objective function is computationally too expensive to be evaluated inside EA. Details on the methods used in this framework, developed at the Division of Fluid Dynamics, Chalmers University of Technology, are discussed. Moreover, the methodology and implementation is validated for a set of well-known benchmark problems and it is finally applied to perform an aerodynamic optimization of a CROR configuration.

III. Optimization methodology

When using HAMON, two different optimization methodologies can be used depending on the cost of evaluating the objective function(s). In cases when hundreds or even thousands of objective function evaluations are computationally affordable, HAMON evaluates every new individual of the EA by directly computing its objective function(s) value(s). On the other hand, if the computational cost of evaluating each design is considerably expensive (such as a 3D RANS CFD simulations), meta-modeling can be used. In this case, a strategy similar to the one used by Ellbrant [8] is adopted, which consists of five main parts as illustrated in Fig. 1, namely, initial design set, geometry and mesh generation, design evaluation, meta-model construction and optimization by means of an EA.

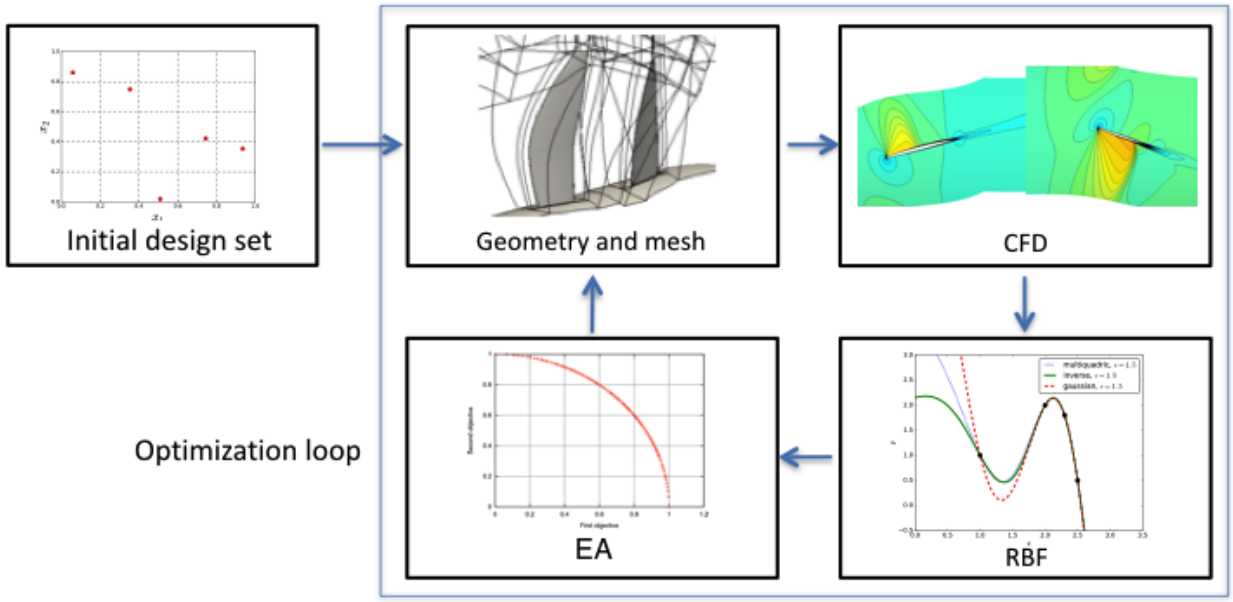


Fig. 1 Optimization procedure implemented in HAMON when RBFs are used.

In the first step of the workflow a Latin Hypercube Sampling (LHS) is performed to get a set of initial designs. With the initial design set generated, the optimization loop is started by generating a geometry and a mesh for each of the designs and then evaluating them by performing 3D RANS CFD simulations. Once all the initial designs have been evaluated, the obtained data set is passed to the meta-model where one response surface per objective function is

constructed. These response surfaces are fed to the evolutionary algorithm to be used as approximation models, thus allowing to avoid direct evaluation of each of the designs during the EA run. After the optimization process, a set of optimum designs is obtained. The credibility that the obtained set of designs is the true optimal, is directly related to the accuracy of the response surfaces. Therefore, in order to enhance the prediction capabilities, more information needs to be provided to the meta-model, and to do so, some of the found optimum designs are selected and one more iteration in the optimization loop is performed. The geometry and mesh for each of the selected individuals is generated and their objective functions evaluated by means of CFD simulations. The newly obtained data is combined with the previously available one for the new response surfaces to be constructed and passed to the EA. After a new optimum design set is found, some more individuals are selected so that the optimization loop can be restarted. The optimization procedure is repeated until the response surface is considered converged or satisfactory designs have been found.

A. Latin Hypercube Sampling

Latin hypercube is a sampling technique that falls in the category of design of experiments, which randomly spreads a certain amount of samples, m, in an n-dimensional design space [9]. Each design variable's range is divided into m equal intervals and the LHS method guarantees that only one design lays in each of the intervals of each variable. Figure 2 illustrates an example for a 2-dimensional space where five samples are required, with the variables $x_1, x_2 \in [0, 1]$.

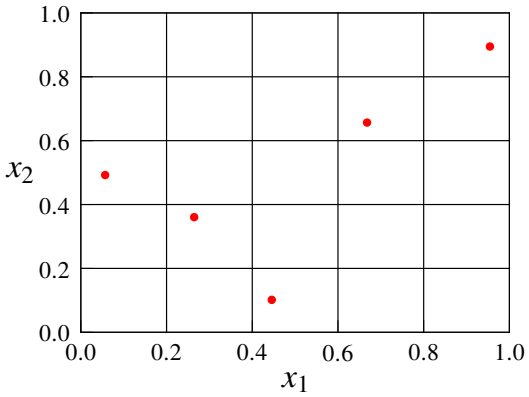


Fig. 2 Example of LHS in 2D space with five samples.

B. Geometry and mesh generation

As previously mentioned, HAMON has been used to perform an aerodynamic optimization of a CROR. This is done by keeping a constant hub contour, spacing between both rotors and rear blade clipping, so that only the rotor blades are optimized. Some relevant design parameters are presented in Table 1.

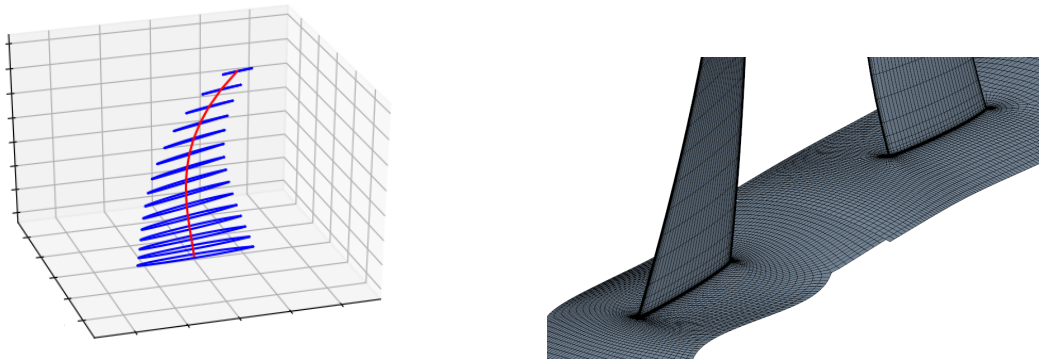
Table 1 CROR design parameters.

Configuration	12x10 blades
D_1	4.26 m
D_2	3.84 m
Hub to tip ratio (forward rotor)	0.35
Rotor-rotor spacing	$0.27D_1$
Airfoil	NACA 16

To be able to apply the optimization procedure to the CROR case, its geometry needs to be parametrized by a finite number of variables which are handled by the EA. The blade profiles used for constructing the blades in this work belong to the NACA 16 family of airfoils with a rounded trailing edge. These airfoils are defined by four variables, namely camber, thickness, chord and an angle of attack with respect to the undisturbed flow. The first two are used in

order to define the shape of the profile, which is later scaled with the chord, and finally pitched by rotating it according to the defined angle of attack. In order to define all the airfoil profiles conforming a blade, the four aforementioned variables are defined at four different radial positions (hub, 60% span, 85% span, and tip) and their corresponding radial distributions are constructed using B-splines. Similarly, variables describing the stacking points at the same radial positions are used to define a Bézier curve to generate the complete stacking line. The blades are finally constructed by stacking a set of airfoils on cylindrical surfaces with respect to their center of mass at their corresponding stacking points. This resulted in a total of 44 variables for defining both the front and the rear rotor. An example of a generic blade constructed using the method just described together with the stacking line for one blade is shown in Fig. 3 a).

The optimization method described in this paper, when applied to the CROR case, uses CFD simulations as the design evaluation method, thus requiring a computational mesh that represents both rotor blades geometries. The mesh is generated using AutoGrid5™ by NUMECA. Due to the large amount of CFD simulations that are required during the optimization process, the cell count as been kept to approximately 1.7 million cells per configuration. An example of the mesh for one of the configurations is shown in Fig. 3 b).



a) Stacking procedure. **b) Computational mesh.**

Fig. 3 Generic CROR configuration geometry and mesh.

C. Computational fluid dynamics

The commercial software STAR-CCM+ v12.04 is used in order to solve the compressible Navier-Stokes equations and predict the performance of each design. The designs are evaluated by steady state simulations at top of climb conditions with the coupled-implicit solver available in STAR-CCM+. Turbulence is modeled using the one equation Spalart-Allmaras turbulence model [10] and the "all y^+ wall treatment" available in STAR-CCM+ is chosen to model the flow close to the walls. All the simulations were run for 5000 iterations, which was found to be enough for the solver to reach a converged solution in several test simulations run prior to the optimization. Since the optimization process depicted in Fig. 1 is unsupervised, and not all the simulations are checked, a criteria that the root-mean-square deviation of the objective function values taken over the last 1000 iterations has to be smaller than 0.1% is defined. If the criteria is fulfilled, the simulation is considered converged, and the corresponding data is added to the database for the RBF construction, otherwise the simulation is discarded.

A schematic view of the computational domain with a summary of the boundary conditions used can be seen in Fig. 4. The domain extends 3.5 front rotor diameters from the rotational axis in height and 5.5 in total length. Both rotating domains are connected using a mixing plane and only one blade per row is simulated using periodic boundary conditions at the pitch-wise boundaries.

The thrust coefficient (C_T) and efficiency (η) are used as objective functions for the optimization. After each CFD simulation is completed the value of the these two objective functions is extracted and added to the database. These objective functions are computed as follows,

$$\eta = \frac{TV_\infty}{P} \tag{1}$$

$$C_T = \frac{T}{\rho_\infty n_1^2 D_1^4} \tag{2}$$

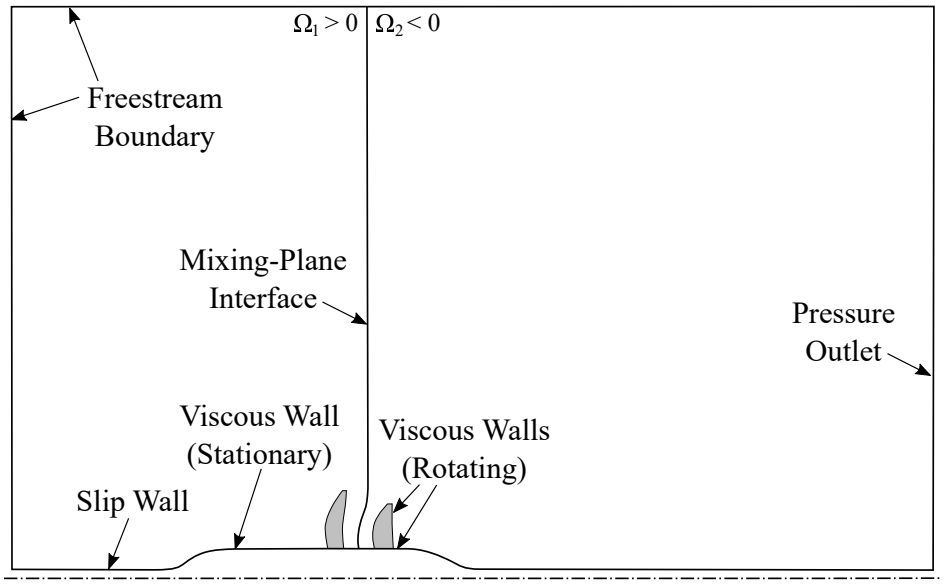


Fig. 4 A schematic overview of the computational domain including boundary conditions. Both blades and corresponding hubs rotating.

where V_∞ represents the free-stream velocity, P is the total power (accounting for both blades and their corresponding rotating hubs), T is the total thrust generated by the entire engine when accounting for nacelle drag, ρ_∞ is the free-stream density and n_1 is the rotational speed of the front blade.

D. Meta-modeling

Evolutionary algorithms generally require more design evaluations than other conventional optimization methods. This, together with the fact that CFD simulations are computationally expensive and time consuming, makes it prohibitive to directly evaluate each individual during the EA optimization. Instead, a meta-model is used to approximate the performance of each individual. Meta-models do this by fitting a response surface to a set of designs whose performance is known, so that the performance of new designs can be estimated without the need of a CFD simulation. HAMON uses RBFs for meta-modeling.

A RBF is an approximation method where the estimated value of an unknown design is a function of the Euclidean distance in the design space to a set of known points [11]. Let $\mathbf{x_i}$ be the design variables of the already evaluated data set, $y(\mathbf{x})$ their evaluated value, $\mathbf{x}$ the design variables of the case that needs to be estimated, and $\hat{y}(\mathbf{x})$ its predicted value. The approximation made by the RBF can then be expressed as

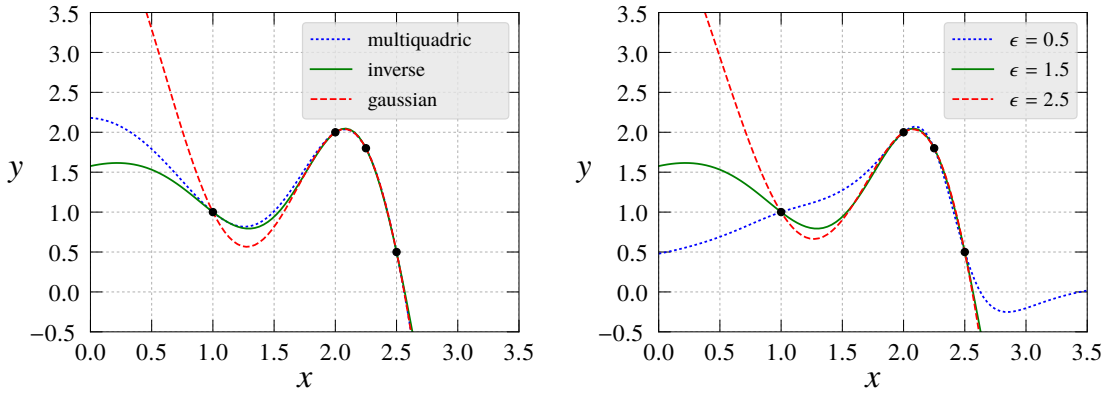
$$\begin{aligned} \hat{y}(\mathbf{x}) &= \sum_{i=1}^{N} w_i \phi(r_i) \\ r_i &= ||\mathbf{x} - \mathbf{x}_i|| \end{aligned} \tag{3}$$

where N is the number of points in the known data set, r_i is the Euclidean distance in the design space and $\phi(r)$ is the basis function. Several different basis functions are available, some common ones being

$$\begin{aligned} \phi(r) &= \sqrt{\left(\frac{r}{\epsilon}\right)^2 + 1} \\ \phi(r) &= \frac{1}{\sqrt{\left(\frac{r}{\epsilon}\right)^2 + 1}} \\ \phi(r) &= e^{-\left(\frac{r}{\epsilon}\right)^2} \end{aligned} \tag{4}$$

These are often referred to as multiquadric, inverse and gaussian respectively. In Eq. (4), ϵ is a parameter that needs to be tuned in order to have the best possible representation of the real response surface. Finally, the weights w_i in Eq. (3) are calculated to ensure that $\hat{y}(\mathbf{x}_i) = y(\mathbf{x}_i)$ is satisfied.

Every time a new design is evaluated and included in the database, the best basis function and ϵ combination to construct the response surface is recomputed. This tuning is done by using a percentage of the available database to construct the RBF. The performance of the remaining designs of the database (whose performance is known since they have already been evaluated with CFD) is predicted with the RBF, and the estimation compared with the real value from the simulations. This allows to measure the prediction capabilities of the RBF in order to tune the basis function and ϵ parameter. Figure 5 illustrates how the choice of basis function and ϵ affects the shape of the response surface for a simple 2D case where 4 designs are known.



a) Effect of basis function (all $\epsilon = 1.5$). **b) Effect of ϵ (all RBFs are of type inverse).**

Fig. 5 RBF parameter choice, black dots represents the known points.

E. Evolutionary algorithms

As previously mentioned, HAMON uses evolutionary algorithms as its optimization method, GA and DE being implemented, both of which can handle single- and multi-objective, constrained and unconstrained optimization problems. These two evolutionary algorithms are stochastic optimization methods inspired by biological evolution and share some common features; they are population-based algorithms and use similar mechanisms during the optimization process; selection, mutation and recombination [12].

In EA terms, a candidate with its corresponding design variables is called an individual, and the group of all individuals is referred to as a population. For GAs, every individual stores each design variable encoded in genes inside its chromosome. The GA implemented in HAMON uses binary encoding, which means that a design variable is defined by a set of ones and zeros (each one or zero is called a gene), and a chromosome is just a collection of the genes encoding all the design variables. On the other hand, in the DE implementation the design variables are stored in parameter vectors, where the design variables are not encoded but instead stored with their actual value. In both the DE and GA implementation the initialization of the first generation is done in a random manner. The main difference between single- and multi-objective optimization problems occurs when trying to compare two individuals. In the case of single-objective optimization, it is straightforward to get the best individual when comparing two of them, whereas in multi-objective problems, this becomes a more complicated process. In the following subsections, the GA and DE algorithms available in HAMON are explained in greater detail.

1. Single-objective genetic algorithm

For single-objective optimization problems, HAMON uses a standard genetic algorithm process, divided into four main steps, for advancing the population to the next generation:

1) **Selection** (tournament selection of size two): is based on the principle "survival of the fittest". A pair of individuals is selected randomly from the population and compared. After comparing them, the strongest

individual (the one whose objective function value is better) survives with a user-defined probability, called tournament selection parameter, otherwise, the weakest one lives on.

2) **Crossover** (one-point crossover): this operation mirrors the reproductive process seen in nature. Every time two individuals have been selected through tournament selection, called parents, they get the chance to reproduce with a user-defined probability. If it turns out that they will produce an offspring, a random point is selected in their chromosomes and the two sides divided by that point are swapped between the parents in order to produce two children (as illustrated in Fig. 6). In case of an offspring, the two children survive. If the children do not survive, the two parents will.

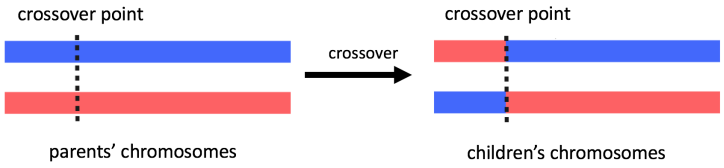


Fig. 6 Illustration of one-point crossover.

3) **Mutation**: in this operation, every gene in the chromosomes of the two outcome individuals from the crossover process is swapped with a user-defined probability.
4) **Elitism**: by using this technique the algorithm guarantees that the best individual is not lost or worsened through the above-mentioned processes. Before the new generation's population is constructed, a copy of the best individual is saved and is used to replace a random individual after the new population has been obtained.

These processes are performed in a loop until a stopping criteria is fulfilled; which usually is that a fixed number of generations have been evaluated.

2. Multi-objective genetic algorithm

One of the main differences between single- and multi-objective optimization is that the solution to multi-objective problems is not a single individual but instead, a set of them called the pareto-front. For dealing with multi-objective optimization problems there are different GA algorithms available, among which the NSGA-II algorithm by Deb et al. [3] is implemented in HAMON. As previously mentioned, direct comparison between individuals is not straightforward since one individual could perform better in one of the objective functions, but worse on another one. In order to address this issue, NSGA-II introduced the dominance operator ($\prec$): Let N be the number of objective functions, $i1$ and $i2$ the two individuals to be compared and $of_m(j)$ the value of the mth objective function of individual j, then individual $i1$ dominates individual $i2$ if:

$$i1 \prec i2 \leftrightarrow of_m(i1) \leq of_m(i2),\ \forall m \in \{1, ..., N\} \ \wedge\ \exists k \in \{1, ..., N\} \ :\ of_k(i1) < of_k(i2) \tag{5}$$

Equation (5) is formulated here for the case when all the objective functions are to be minimized, if that is not the case, $<$ and $\leq$ change accordingly. The dominance operator is used to classify the population into different ranks, where rank one is the best. These are the individuals that are not dominated by any other individuals in the population, whereas individuals that are only dominated by rank one members are rank two and so on.

The crowded distance parameter (cd) is another new concept introduced in NSGA-II. It aims to quantify the density of the surroundings of an individual in the objective functions space as follows: Let $\mathcal{A}_m$ be the set containing individuals with a common rank sorted according to the value of their objective function m, M the size of $\mathcal{A}_m$ and of_m^{max} and of_m^{min} the maximum and minimum values of the objective function m in that set respectively. The crowded distance for each of the individuals i , $cd(i)$, belonging to the set is calculated as follows,

$$\begin{aligned}
&\text{for each i, set } cd(i) = 0 \\
&\text{for } m = 1 \text{ to } N \\
&\quad cd\big(\mathcal{A}_m(1)\big) = cd\big(\mathcal{A}_m(M)\big) = \infty \\
&\quad \text{for } i = 2 \text{ to } (M-1) \\
&\qquad cd\big(\mathcal{A}_m(i)\big) = cd\big(\mathcal{A}_m(i)\big) + \Big[of_m\big(\mathcal{A}_m(i+1)\big) - of_m\big(\mathcal{A}_m(i-1)\big)\Big] / \big(of_m^{max} - of_m^{min}\big)
\end{aligned} \tag{6}$$

The introduction of the dominance operator, rank, and crowded distance concepts allows individuals to be directly compared as follows. First, the rank of both individuals are compared, if they are different, the one with the lowest ranking is considered to be the best. On the other hand, if both have the same ranking, then the one with the largest crowded distance is selected as the best candidate, thereby rewarding the individuals that lay in the less explored areas.

Most of the processes in NSGA-II are done in same manner as in the single-objective optimization; tournament selection of size two (with the explained difference when comparing individuals), and mutation and crossover are performed in the exact same way. Since there is no way of selecting an absolute best individual for saving it for the next generation, the elitism process in done differently. NSGA-II applies an elitism process that involves the newly generated population through mutation, crossover and selection. This newly generated population is here called Q, and the old population, P. Once the new population Q is obtained from advancing population P one generation, both of them are combined in a population of twice the size, PQ. The individuals in this new population are ranked and their crowded distances computed. The population P of the new generation (which is of the same size as the original one) is obtained by the process illustrated in Fig. 7. First, rank one individuals of population PQ are introduced in the new population P. Later on, the individuals of rank 2 are added and so on. This process proceeds until adding all the individuals of a certain rank will max out the size of the new population P of the next generation. Then, only the individuals with larger crowded distance are added until the new population is of the right size and the rest of the individuals are discarded.

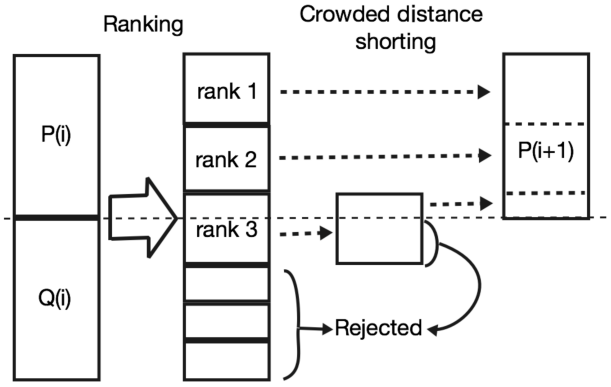


Fig. 7 Elitism process in the NSGA-II algorithm. Here i represents the generation number of the population.

3. Differential evolution

Differential evolution is an stochastic optimization method that lies in the category of evolutionary algorithms [2]. The main procedure for both single- and multi-objective optimization as implemented in HAMON is practically the same as for the GA, but mutation, recombination (analogous to crossover) and selection are performed differently. The main processes in DE are explained here with a notation similar to the one used by Rai [13]. As previously mentioned, in DE an individual has no chromosome, but instead a parameter vector:

$$\mathbf{X_{i,n}} = \left[x_{1,i,n}, x_{2,i,n}, \ldots, x_{D,i,n}\right]$$

where $\mathbf{X_{i,n}}$ is the parameter vector of the i^{th} individual in a M size population of the n^{th} generation, D is the number of variables and $x_{j,i,n}$ is the j^{th} design variable of that individual.

To evolve $\mathbf{X_{i,n}}$ to the next generation's $\mathbf{X_{i,n+1}}$, the following three processes are applied:

1) **Mutation**: randomly select three individuals, $\mathbf{X_{a,n}}$, $\mathbf{X_{b,n}}$ and $\mathbf{X_{c,n}}$ from the population with $a \neq b \neq c \neq i$, then a new individual, $\mathbf{W}$, is created as:

$$\mathbf{W} = \left[w_1, w_2, \ldots, w_D\right] = \mathbf{X_{a,n}} + F(\mathbf{X_{b,n}} - \mathbf{X_{c,n}}) \tag{7}$$

where $0 < F < 1$ is a user define parameter. In the case that any of the design variables falls out of the allowed range, it is adjusted.

2) **Recombination**: the newly created individual (**W**) is recombined with the original one ($\mathbf{X_{i,n}}$) to generate a new individual (**Z**) as:

$$\mathbf{Z} = \left[z_1, z_2, ..., z_D\right], \text{ where } z_j = \begin{cases} w_j & \text{if } r_j \leq C \\ x_{j,i,n} & \text{if } r_j > C \end{cases} \tag{8}$$

where r_j is uniformly distributed random number, and $0 < C < 1$ a user defined parameter.

3) **Selection**: finally, the individual of the new generation $\mathbf{X_{i,n+1}}$ is a copy of the best individual when comparing **Z** and $\mathbf{X_{i,n}}$. This is only done in single-objective optimization and it guarantees that the best individual does not become worse from one generation to the next. The reason why selection is not performed in multi-objective problems is that the process described before and illustrated in Fig. 7 already takes care of comparing the newly generated individuals and those from the previous generation.

IV. Validation

In this section HAMON is applied to several benchmark cases on different types of optimization problems and the obtained results are discussed.

A. Constrained single-objective optimization

In order to test the capabilities of HAMON handling single-objective optimization problems with constraints, Golinski's speed reducer problem is chosen [14], which aims to minimize the following function,

$$f(\mathbf{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \tag{9}$$

with the variable ranges shown in Table 2, x_3 being a variable that only can take on integer values and subjected to the inequality constraints defined in Table 3.

Table 2 Variable ranges for Golinski's speed reducer problem.

i	1	2	3	4	5	6	7
x_i range	[2.6, 3.6]	[0.7, 0.8]	[17, 28]	[7.3, 8.3]	[7.3, 8.3]	[2.9, 3.9]	[5.0, 5.5]

Table 3 Constraints for Golinski's speed reducer problem.

Function	Constraint
$g_1(\mathbf{x})$	$27 - x_1x_2^2x_3 \leq 0$
$g_2(\mathbf{x})$	$397.5 - x_1x_2^2x_3^2 \leq 0$
$g_3(\mathbf{x})$	$1.93x_4^3 - x_2x_3x_6^4 \leq 0$
$g_4(\mathbf{x})$	$1.93x_5^3 - x_2x_3x_7^4 \leq 0$
$g_5(\mathbf{x})$	$\left[\left(745x_4/(x_2x_3)\right)^2 + 16.9 \cdot 10^6\right]^{0.5} - 110x_6^3 \leq 0$
$g_6(\mathbf{x})$	$\left[\left(745x_5/(x_2x_3)\right)^2 + 157.5 \cdot 10^6\right]^{0.5} - 85x_7^3 \leq 0$
$g_7(\mathbf{x})$	$x_2x_3 - 40 \leq 0$
$g_8(\mathbf{x})$	$5x_2 - x_1 \leq 0$
$g_9(\mathbf{x})$	$x_1 - 12x_2 \leq 0$
$g_{10}(\mathbf{x})$	$1.5x_6 - x_4 + 1.9 \leq 0$
$g_{11}(\mathbf{x})$	$1.1x_7 - x_5 + 1.9 \leq 0$

In order to solve this problem, single-objective differential evolution with 50 individuals was run for 200 generations with both the F and C parameters (see Eqs. (7) and (8)) set equal to 0.5. The evolution of the optimization process over the generations can be seen in Fig. 8, where the best found individual of the last generation had a fitness value of $f(\mathbf{x}) = 2994.34150$, which agrees very well with results obtained by other authors [13–15]. A summary of the

optimized design including values of the constraints, is reported in Table 4. These results show that the optimized design fulfills all eleven constraints in Table 3.

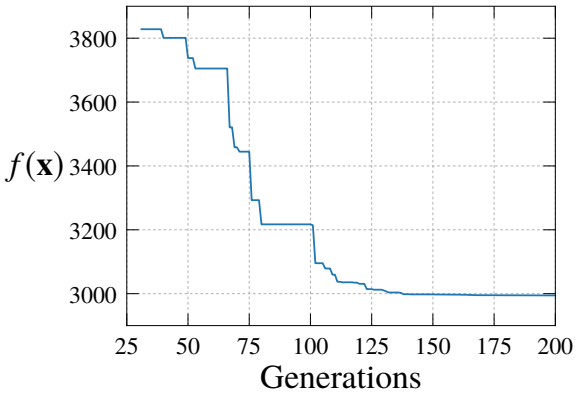


Fig. 8 Best found fitness in each generation for Golinski's speed gear problem.

Table 4 Golinski's speed reducer best found individual summary.

i	x_i	$g_i(\mathbf{x})$
1	3.5000000511	−2.15500
2	0.7000000042	−98.13501
3	17	−748.32008
4	7.3000035467	−8409.07566
5	7.7153239259	−0.00032
6	3.3502147609	$-3.76243 \cdot 10^{-5}$
7	5.2866544716	−28.09999
8		$-3.00999 \cdot 10^{-8}$
9		−4.89999
10		−0.37468
11		$-4.00713 \cdot 10^{-6}$

B. Unconstrained multi-objective optimization

HAMON has also been tested on two well known multi-objective optimization benchmark cases using both GA and DE. Zitzler et al. [16] defined a set of multi-objective functions that have been widely used for EA validation and performance assessment, also often referred to as the ZDT functions [3, 17]. The implementation has been tested for the ZDT 1 and ZDT 3 functions, which are described in Table 5.

In both test cases DE is run for 200 generations and GA for 300. DE is run using 20 and 30 individuals for ZDT 1 and ZDT 3 respectively and GA using 50 and 100. The results obtained are shown in Fig. 9, where good agreement between the pareto-front obtained with both optimization methods and the real one can be seen.

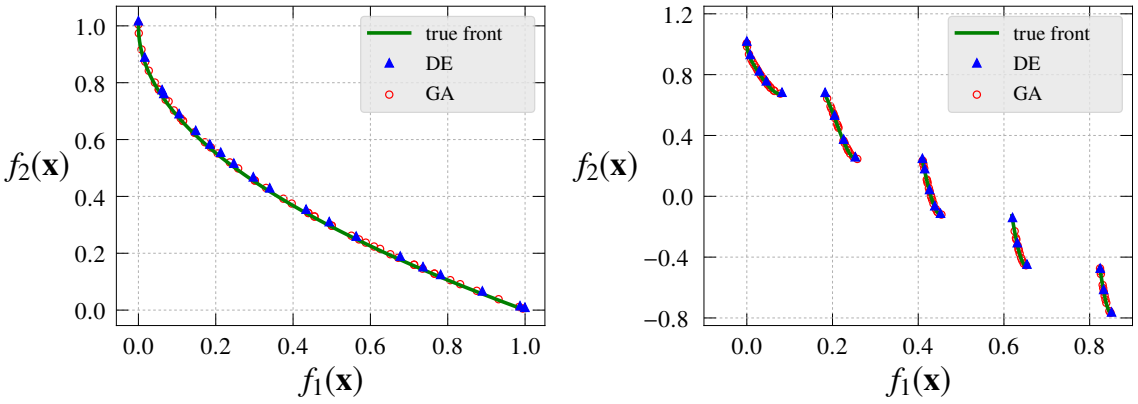
C. Constrained multi-objective optimization

In order to test HAMON's ability to deal with constrained multi-objective problems, it is tested on the TNK function, which is a two dimensional problem where two objective functions are to be minimized subject to two constraints (see Table 6).

The results of the optimization of the TNK function using DE with 60 individuals and 200 generations, with both F and C parameters (see Eqs. (7) and (8)) equal to 0.5, can be seen in Fig. 10. The found pareto-front agrees well with

Table 5 Definition of the ZDT 1 and ZDT 3 functions.

Function	variables, m	Variable range	Functions to minimize ($f_1(\mathbf{x})$ & $f_2(\mathbf{x})$)
ZDT 1	30	[0, 1]	$f_1(\mathbf{x}) = x_1$
			$f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{x_1/g(\mathbf{x})}\right]$
			$g(\mathbf{x}) = 1 + 9\left(\sum_{i=2}^{m} x_i\right)/(m-1)$
ZDT 3	30	[0, 1]	$f_1(\mathbf{x}) = x_1$
			$f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{x_1/g(\mathbf{x})} - \frac{x_1}{g(\mathbf{x})}\sin(10\pi x_1)\right]$
			$g(\mathbf{x}) = 1 + 9\left(\sum_{i=2}^{m} x_i\right)/(m-1)$



a) ZDT 1. **b) ZDT 3.**

Fig. 9 Unconstrained multi-objective validation.

Table 6 TNK function definition.

Functions	Constrains	Variables range
$f_1(\mathbf{x}) = x_1$	$g_1(\mathbf{x}) = -x_1^2 - x_2^2 + 1 + 0.1\cos(16\arctan(x_1/x_2)) \leq 0$	$x_1 \in [0,\ \pi]$
$f_2(\mathbf{x}) = x_2$	$g_2(\mathbf{x}) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 - 0.5 \leq 0$	$x_2 \in (0,\ \pi]$

results reported by other authors [3, 18].

D. Multi-objective optimization with the use of RBFs

Lastly, the optimization process depicted in Fig. 1, where RBFs are used, is tested on a multi-objective unconstrained problem, using again the ZDT 1 function previously defined in Table 5. The process is started by generating a LHS containing 250 samples which are evaluated. With the resulting data from the LHS evaluation the RBF is constructed. This radial basis function is fed into the DE algorithm (that runs with the same settings as the ones presented in Sec. IV B), which selects 25 among the best found individuals to be evaluated. These are added to the previous database, so that a new RBF with more information can be constructed. This process is repeated a total of 6 times. Figure 11 shows how the DE algorithm approaches the real pareto-front as several optimization loops are run, bringing the RBF closer to convergence. Note that the pareto-fronts presented in Fig. 11 are the ones obtained from the optimization made by the DE algorithm using the RBF, not from individuals evaluated with the analytical expression for the ZDT 1 function. This explains why in Fig. 11 b) the pareto-front of the sixth loop is showing values which are lower than the true minimum ones.

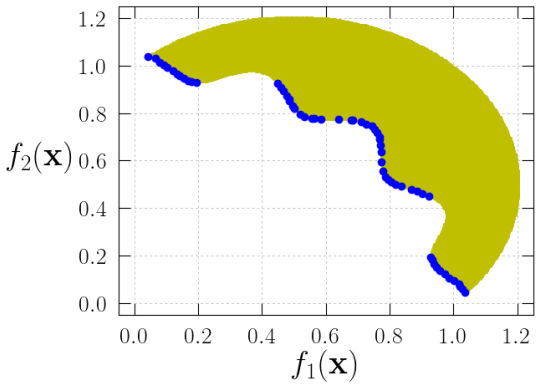
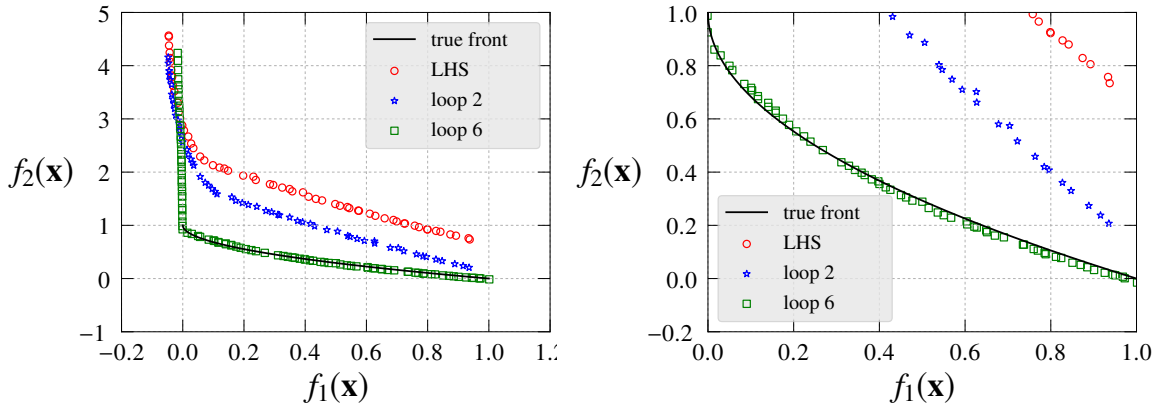


Fig. 10 Validation of the implementation on the TNK function. The yellow shadowed area represents the feasible space and the blue dots the pareto-front found by HAMON.



a) entire pareto-front **b) close up view**

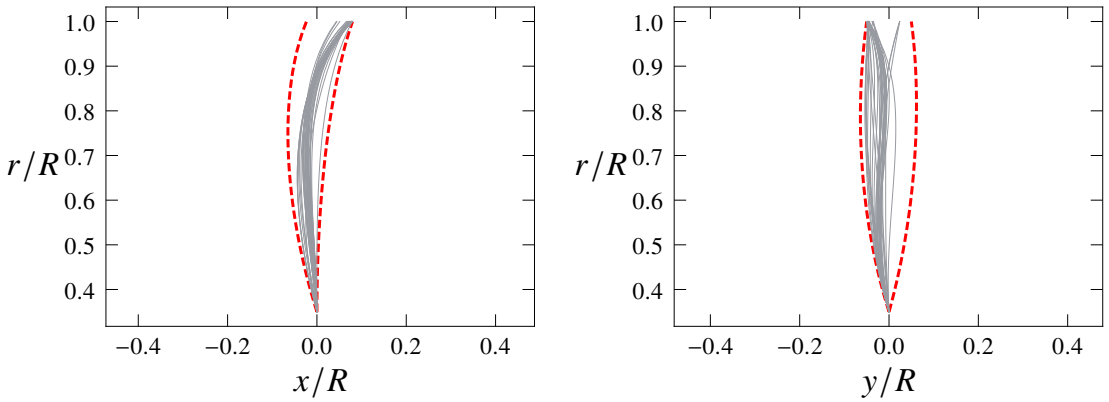
Fig. 11 Evolution of the DE over several optimization loops.

V. Results from the CROR optimization

The methodology explained in Sec. III, has been applied to maximize the efficiency and the thrust coefficient for a CROR at top of climb conditions. A total of 44 variables are used to parametrize the complete geometry of a configuration where the hub, blade spacing and rear blade clipping are kept constant for simplicity. Figure 12 provides an example on the different possible stacking lines of the front rotor during the optimization. The red dashed lines represent the limits defined by the variable ranges, and the gray solid lines the stacking lines of all the configurations that belong to the converged pareto-front obtained from the optimization.

Initially a LHS with 900 designs was evaluated from which the first response surfaces were constructed. As previously discussed, special attention has been put into making sure that any design which is added to the database has converged to the required criteria, in order not to contaminate the RBF data by introducing values from non-converged simulations. Multi-objective differential evolution is used as the optimization method, using a population size of 200 that is run for 300 generations. These numbers are probably on the conservative side when it comes to what is actually needed to converge the optimization. Overruning the optimizer is however not a problem, since the time consuming part of this optimization process is the design evaluation by means of CFD.

During the entire optimization process 75 optimization loops are run, and in every run, 10 individuals are selected from the pareto-front to be evaluated by CFD. These 75 loops combined with the initial LHS resulted in a total of 1650 CFD evaluations. During the first 35 optimization loops, the individuals are selected from the pareto-front by searching for those whose design space area is less crowded. This is done comparing both the individuals from the last generation

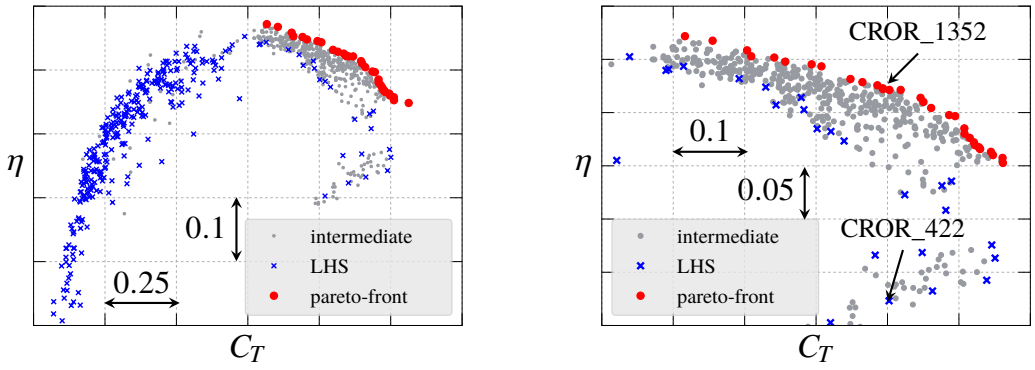


a) Meridional plane, flow direction is from left to right.

b) Azimuthal-radial plane, blade rotating counter-clockwise.

Fig. 12 Design possibilities of the front rotor stacking line. — — encloses the allowed design space and stacking lines from the designs belonging to the pareto-front.

obtained from DE and also the database of previously evaluated designs. The purpose of this is to explore the design space so that a good spread of samples are available for the RBFs construction. During the rest of the optimization loops, the individuals are selected based on the crowdedness on the objective function space instead, aiming to achieve a pareto-front as continuous as possible. Figure 13 presents the results of the optimization process in the objective function space, all those been results obtained by CFD simulations, not from the RBF prediction. It can clearly be seen that the optimization has been able to push the CROR configurations towards higher efficiencies and thrust coefficients as compared to the initial LHS.



a) All evaluated configurations.

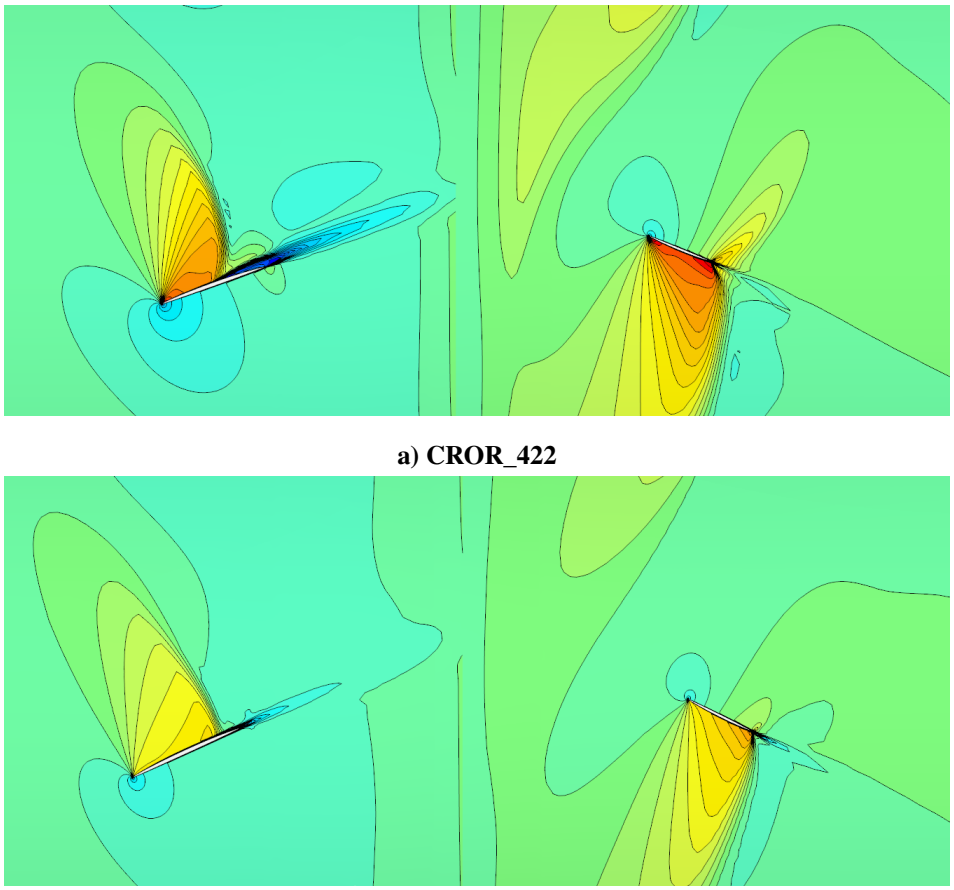
b) Close-up view of the pareto-front.

Fig. 13 Optimization result.

To see how HAMON has managed to improve the CROR configuration, one design from the LHS and one from the pareto-front with nearly equal thrust coefficients have been selected. This chosen thrust coefficient is the one that corresponds to the aimed thrust target for the top of climb conditions. The performance relative to the rest of the evaluated designs of the two selected configurations can be seen in Fig. 13 b), where CROR_422 is the design from the LHS and CROR_1352 the one from the pareto-front.

Figure 14 shows the relative Mach number contours for both selected configurations at 75% span of the front rotor. It can clearly be seen how HAMON has managed to optimize the blades so that the shock formed at the suction side of

both rotors becomes weaker. It can also be noted that not only the strength of the shocks has been decreased but also the placement of the front rotor shock has been pushed further back towards the trailing edge. This helped to considerably reduce the separation that can be seen there and thereby decrease the associated losses.



a) CROR_422

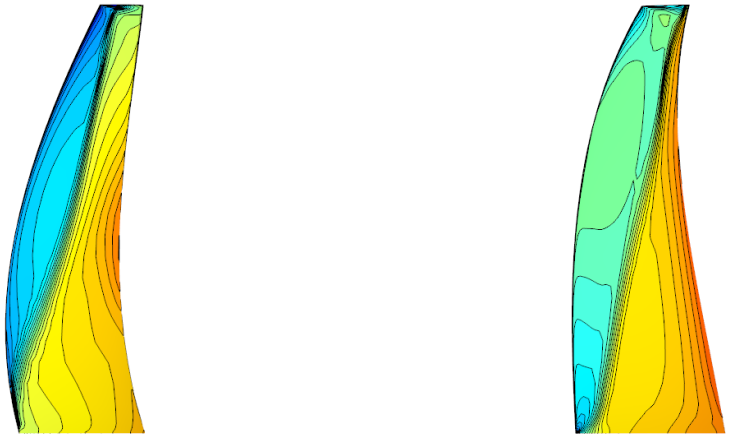
b) CROR_1352

Fig. 14 Relative Mach number contours at 75% span of the front rotor (same scale used). Flow from left to right.

Figure 15 shows the static pressure contours for both selected configurations on the suction side of the front rotor. By looking at these it can also be seen that the shock wave has been pushed downstream and is situated closer to the trailing edge, specially at higher radii. This together with the reduction of the shock strengths mentioned before, are some reasons for the increase in efficiency.

VI. Conclusions

An automated platform for turbomachinery design and optimization based on evolutionary algorithms, called HAMON, which can handle singe- and multi-objective, as well as constrained and unconstrained optimization problems has been presented and the main components of its implementation discussed. A validation of the implementation of HAMON has been carried out in which some of its capabilities have been tested, namely; single-objective constrained optimization, both constrained and unconstrained multi-objective optimization and unconstrained multi-objective optimization using RBFs. The results for all the validation cases have been satisfactory, showing good agreement with their corresponding analytical solution and results found by other authors. Furthermore, the presented methodology where RBFs are used has been applied to optimize a CROR configuration aiming to maximize two objective functions, namely thrust coefficient and efficiency. This later optimization has shown satisfactory results where the designs from the LHS have been pushed towards configurations with higher thrust and efficiency. Two designs with similar thrust



a) CROR_422 **b) CROR_1352**

Fig. 15 Static pressure contours on the front blade suction side (same scale used). Flow from left to right.

coefficient, one belonging to the LHS and the other one to the pareto-front found after the optimization process have been analyzed and compared.

Acknowledgements

The authors would like to acknowledge the Swedish National Infrastructure for Computing (SNIC) for providing computer resources at the High Performance Computing Center North (HPC2N) in Umeå, Sweden and Chalmers Centre for Computational Science and Engineering (C^3SE) in Gothenburg, Sweden.

References

[1] Holland, J. H., "Adaptation in Natural and Artificial Systems," *University of Michigan Press*, 1975.

[2] Storn, R., and Price, K., "Differential Evolution–a Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces," *Journal of Global Optimization*, Vol. 11, No. 4, 1997, pp. 341–359.

[3] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *Evolutionary Computation, IEEE Transactions on*, Vol. 6, No. 2, 2002, pp. 182–197.

[4] Seada, H., and Deb, K., "A Unified Evolutionary Optimization Procedure for Single, Multiple, and Many Objectives," *IEEE Transactions on Evolutionary Computation*, Vol. 20, No. 3, 2016, pp. 358–369.

[5] Ellbrant, L., Eriksson, L.-E., and Martensson, H., "Design of Compressor Blades Considering Efficiency and Stability Using CFD Based Optimization," *Proceedings of ASME Turbo Expo*, Vol. 2012, 2012.

[6] Marinus, B., Roger, M., Van den Braembussche, R., and Bosschaerts, W., "Multidisciplinary Optimization of Propeller Blades: Focus on the Aeroacoustic Results," *17th AIAA/CEAS Aeroacoustics Conference (32nd AIAA Aeroacoustics Conference)*, 2011, p. 2801.

[7] Schnell, R., Yin, J., Voss, C., and Nicke, E., "Assessment and Optimization of the Aerodynamic and Acoustic Characteristics of a Counter Rotating Open Rotor," *Journal of Turbomachinery*, Vol. 134, No. 6, 2012, p. 061016.

[8] Ellbrant, L., Eriksson, L.-E., and Mårtensson, H., "Balancing Efficiency and Stability in the Design of Transonic Compressor Stages," *ASME Turbo Expo 2013: Turbine Technical Conference and Exposition*, American Society of Mechanical Engineers, 2013, pp. V06BT37A017–V06BT37A017.

[9] McKay, M. D., Beckman, R. J., and Conover, W. J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 42, No. 1, 2000, pp. 55–61.

[10] Spalart, P. R., and Allmaras, S. R., "A One Equation Turbulence Model for Aerodynamic Flows," *30th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA Paper 92-0439, 1992.

[11] Buhmann, M. D., "Radial Basis Functions: Theory and Implementations," *Cambridge Monographs on Applied and Computational Mathematics*, Vol. 12, 2004.

[12] Wahde, M., *Biologically Inspired Optimization Methods: an Introduction*, WIT Press, 2008, Chap. 3, pp. 35–67.

[13] Rai, M. M., "Single-and Multiple-Objective Optimization with Differential Evolution and Neural Networks," *VKI lecture series: introduction to optimization and multidisciplinary design*, Vol. 58, 2006.

[14] Ray, T., "Golinski's Speed Reducer Problem Revisited," *AIAA journal*, Vol. 41, No. 3, 2003, pp. 556–558.

[15] Kodiyalam, S., "Evaluation of Methods for Multidisciplinary Design Optimization (MDO). Phase 1," 1998.

[16] Zitzler, E., Deb, K., and Thiele, L., "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, Vol. 8, No. 2, 2000, pp. 173–195.

[17] Chase, N., Rademacher, M., Goodman, E., Averill, R., and Sidhu, R., "A Benchmark Study of Multi-Objective Optimization Methods," *BMK-3021, Rev*, Vol. 6, 2009.

[18] Zhao, Y., Xiong, S., and Li, M., "Constrained Single- and Multiple-Objective Optimization with Differential Evolution," *Natural Computation, 2007. ICNC 2007. Third International Conference on*, Vol. 4, IEEE, 2007, pp. 451–455.